CIRRELT

Centre interuniversitaire de recherche
sur les réseaux d'entreprise, la logistique et le transport

Interuniversity Research Centre
on Enterprise Networks, Logistics and Transportation

Revue de littérature du problème d'ordonnancement de projet à ressources limitées

Roubila Lilia Kadri
Fayez F. Boctor

Février 2017

CIRRELT-2017-11

Document de travail également publié par la Faculté des sciences de l'administration de l'Université Laval, sous le numéro FSA-2017-002.

Bureaux de Montréal :
Université de Montréal
Pavillon André-Aisenstadt
C.P. 6128, succursale Centre-ville
Montréal (Québec)
Canada H3C 3J7
Téléphone : 514 343-7575
Télécopie : 514 343-7121

Bureaux de Québec :
Université Laval
Pavillon Palasis-Prince
2325, de la Terrasse, bureau 2642
Québec (Québec)
Canada G1V 0A6
Téléphone : 418 656-2073
Télécopie : 418 656-2624

www.cirrelt.ca

Université Laval · McGill · Concordia · ÉTS · UQÀM · HEC Montréal · Polytechnique Montréal · Université de Montréal

Revue de littérature du problème d'ordonnancement de projet à ressources limitées

Roubila Lilia Kadri, Fayez F. Boctor*

Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport (CIRRELT) et Département d'opérations et systèmes de décision, 2325 de la Terrasse, Université Laval, Québec, Canada G1V 0A6

Résumé. Ce document présente une revue de littérature traitant le problème d'ordonnancement de projet à ressources limitées (POPRL), c'est un problème d'optimisation combinatoire de complexité NP-dur (Blazewicz et al., 1983). La résolution d'un POPRL a pour but la détermination des dates d'exécution des tâches en tenant compte des contraintes de préséance et de disponibilité des ressources et ayant comme objectif la minimisation de la durée totale du projet. L'aspect pratique de ce problème dans des contextes industriels divers a conduit à de nombreuses recherches. Les contributions relatives à ce problème portent principalement sur les méthodes exactes de résolution, la détermination de bornes inférieures sur la durée du projet et les méthodes heuristiques (approchées) de résolution. Afin d'avoir une meilleure représentation des situations pratiques, plusieurs généralisations du problème classique ont été proposées. Le problème d'ordonnancement de projet à ressources limitées où les tâches ont plusieurs modes d'exécution (POPRL=PME) est une généralisation du problème qui a eu beaucoup d'intérêt. Un POPRL=PME considère que chaque tâche peut avoir plusieurs modes d'exécution où chaque mode est caractérisé par les quantités des ressources renouvelables et non renouvelables requises et la durée d'exécution si ces quantités de ressources sont utilisées. Il consiste à définir les dates de début ou de fin des tâches, mais aussi à choisir le mode d'exécution à utiliser pour chaque tâche dans l'objectif de minimiser la durée du projet. L'objectif de cette revue est d'effectuer une mise à jour de la littérature disponible concernant le POPRL et le POPRL=PME ainsi que consolider et classifier les travaux de recherche publiés. Et enfin, identifier les perspectives de recherches futures qui ont encore besoin d'être explorées.

Mots-clés. Planification de projet, allocation de ressources, plusieurs modes d'exécution.

Remerciements. Ce projet de recherche a reçu un support financier sous le contrat numéro OPG 0036509 du programme de subventions à la découverte du Conseil de recherches en sciences naturelles et en génie du Canada (CRSNG).

Results and views expressed in this publication are the sole responsibility of the authors and do not necessarily reflect those of CIRRELT.

Les résultats et opinions contenus dans cette publication ne reflètent pas nécessairement la position du CIRRELT et n'engagent pas sa responsabilité.

---

* Auteur correspondant : Fayez.Boctor@cirrelt.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2017

© Kadri, Boctor et CIRRELT, 2017

1 Introduction

Le problème d'ordonnancement de projet à ressources limitées (POPRL) est devenu ces dernières années un problème de référence en gestion de projet (Hartmann (2012)). Il s'agit de déterminer les dates d'exécution des tâches dans l'objectif de minimiser la durée totale du projet, tout en respectant les contraintes de disponibilité des ressources. Trois types de ressources sont identifiés dans la littérature : les ressources renouvelables (ou la quantité disponible est la même à chaque période), les ressources non renouvelables (où la quantité totale que le projet peut utiliser est limitée), et les ressources doublement restreintes (où il y a une limite à la fois sur la quantité utilisable dans chaque période et sur la quantité totale utilisable par le projet).
Le POPRL classique se caractérise habituellement par :

- Plusieurs types de ressources renouvelables qui sont disponibles à chaque période où pour chaque type $k \in K$, une quantité de Q_k unités est disponible à chaque période.
- Un ensemble de tâches noté J, et chaque tâche $i \in J$ a une durée d_i et requiert pour son exécution un nombre r_{ik} d'unités de la ressource de type k; $\forall k \in K$.
- Un ensemble de relation de préséance P, où $(i, j) \in P$ signifie que la tâche j ne peut commencer qu'après la fin de l'exécution de la tâche i.
- Une fois commencée, la tâche ne doit pas être interrompue (préemption non permise).

Deux types de graphes sont utilisés pour représenter un projet, ses tâches et les relations de préséance entre celles-ci. Dans le premier, appelé graphe sommet-tâche, les tâches sont représentées par les sommets du graphe et les arcs indiquent les relations de préséance. Par contre, dans le deuxième, appelé graphe arc-tâche, les arcs représentent les tâches et les sommets indiquent les relations de préséance. Toutes les tâches (arcs) dont le sommet final est le sommet s, sont des prédécesseurs immédiats de toutes les tâches dont s est le sommet initial. Le graphe sommet-tâche est le plus utilisé vu sa simplicité. En plus, représenter un projet par un graphe arc-tâche nécessite souvent l'ajout de tâches fictives.

Cette revue est organisée en 10 sections. La section 2 porte sur la modélisation mathématique du POPRL. La section 3 résume les techniques de determinination des bornes

inférieures sur la durée du projet. Les deux sections 4 et 5 sont consacrées aux méthodes de résolution exactes et heursitiques respectivement. La section 6 donne une classification sommaire de quelques généralisations du POPRL. Les sections 7, 8 et 9 sont consacrées à trois généralisations du problème classique, notamment le problème où les tâches peuvent avoir plusieurs modes d'exécution, le problème avec temps de transfert des ressources entre les sites d'execution des tâches et le problème d'ordonnancement de projet où l'objectif est de minimiser le coût d'exécution du projet. Enfin cette revue se conclut par une discussion sur les avenues de recherches futures qui ont encore besoin d'être explorées.

2 Modélisation

La résolution exacte du POPRL en faisant appel à la programmation linéaire en nombre entier (PLNE) a fait l'objet de plusieurs travaux. De nombreuses formulations mathématiques ont été développées, passant de la formulation en nombre entier de Pritsker et al. (1969), à la formulation de Klein (1999). Voir également le chapitre six du livre de Demeulemeester and Herroelen (2006) et le chapitre trois de Klein (1999) pour plus de détails.
La principale difficulté dans la modélisation des POPRL par la PLNE réside dans la modélisation du problème au moyen d'inégalités linéaires des contraintes de ressources du problème. On distingue principalement les deux types de formulations suivantes :

- **Des formulations à temps continu** : ces formulations introduisent des variables de décision binaires indicées y_{ij} tel que $y_{ij} = 1$, si et seulement si la tâche i précède la tâche j, et 0 sinon. Parmi ces formulations on peut citer celles de Balas (1985), celle de Olaguíbel and Goerlich (1993), et celle d' Artigues et al. (2003).
- **Des formulations à temps discrétisé** : dans ce type de formulation, les variables de décision sont des variables binaires ou entières indicées à la fois par la tâche et la période, noté x_{it}. Parmi celles-ci on cite la formulation classique de Pritsker et al. (1969), de Mingozzi et al (1998), et de Klein (1999).

Deux formulations dans chaque classe sont détaillées dans les quatre paragraphes suivants. Il s'agit de la formulation à temps continu d' Olaguíbel and Goerlich (1993) (voir 2.1), et celle d'Artigues et al. (2003) (voir 2.2). Les formulations à temps discrétisé sont celles de Pritsker et al. (1969) (voir 2.3) et de Mingozzi et al. (1998) (voir 2.4).

2.1 Formulation à temps continu basée sur les ensembles critiques

Pour exprimer les contraintes de ressources dans un POPRL, Olaguíbel and Goerlich (1993) ont intégré la notion d'ensemble critique minimal à la formulation disjonctive de Balas (1985). Un sous-ensemble de tâches C est dit critique si la somme des consommations des tâches de l'ensemble C excède la quantité totale disponible pour au moins une ressource, et il est minimal si aucun sous-ensemble de C n'est critique. On note C_m l'ensemble des sous-ensembles critiques minimaux. Aussi, une paire de tâches i et j est dite incompatible si la somme des ressources requises pour cette paire dépassent la disponibilité pour au moins un type de ressource.

Pour cette formulation nous ajoutons deux tâches fictives notées 1 et F (d'une durée nulle, et qui ne requièrent aucune ressource pour leur exécution) : la tâche 1 précède toutes les tâches sans prédécesseurs et la tâche F est le successeur immédiat de toutes les tâches sans successeurs. On note I l'ensemble des paires de tâches incompatibles.
Le modèle est défini par des variables binaires y_{ij} et des variables continues S_i où :

$$y_{ij} = \begin{cases} 1, & \text{si } i \text{ précède } j \\ 0, & \text{sinon.} \end{cases}$$

S_i : est la date de début de l'exécution de la tâche i, $\forall i \in J \cup \{1, F\}$, $S_i \geq 0$.

Le modèle est donné par :

$$\text{Miminize } S_F \tag{1}$$

Sous les contraintes :

$$y_{ij} = 1 \qquad \forall (i,j) \in P \tag{2}$$

$$y_{ij} + y_{ji} \leq 1 \qquad \forall i,j \in J \cup \{1\} \tag{3}$$

$$y_{ih} \geq y_{ij} + y_{jh} - 1 \qquad \forall i,j,h \in J \cup \{1\} \tag{4}$$

$$y_{ij} + y_{ji} = 1 \qquad \forall (i,j) \in I \tag{5}$$

$$S_j - S_i \geq -M + (d_i + M) y_{ij} \qquad \forall i,j \in J \cup \{1\} \tag{6}$$

$$\sum_{(i,j) \in C \times C} y_{ij} \geq 1 \qquad \forall C \in C_m \tag{7}$$

$$y_{ij} \in \{0,1\} \qquad \forall i,j \in J \cup \{1\} \tag{8}$$

$$S_i \geq 0 \qquad \forall i \in J \cup \{1,F\} \tag{9}$$

La minimisation de la durée totale du projet est donnée par l'expression (1), les contraintes (2) expriment les relations de préséance initiales du problème. Les contraintes

(3) imposent que toutes les paires de tâches (i, j), soient tel que i précède j, ou bien j précède i, ou i et j s'exécutent en parallèle. Les contraintes (4) sont des contraintes de transitivité qui conjointement avec les contraintes (3), éliminent les cycles dans le graphe. Les contraintes (5) imposent que la relation entre chaque paire de tâches incompatibles (i, j) soit tel que i précède j ou l'inverse. Les contraintes (6) lient les deux types de variables du modèle : pour une valeur suffisamment grande de M, les contraintes imposent que si $y_{ij} = 1$, l'exécution de la tâche j doit commencer après la date de fin de la tâche i. L'ensemble des contraintes (7) imposent qu'au moins une contrainte de préséance soit créee dans chaque ensemble critique minimal.

2.2 Formulation à temps continu basée sur les flux

Toujours inspiré par les travaux de Balas (1985) et sur la base de la formulation d'Olaguíbel and Goerlich (1993), Artigues et al. (2003) ont opté pour l'utilisation du concept des flux dans les réseaux pour la modélisation des besoins en ressources au lieu du concept des ensembles critiques. Ce modèle définit les trois types de variables suivants :

- Variables continues usuelles de date de début S_i (pour définir la date de début de chaque tâche du projet)
- Variables séquentielles binaires y_{ij} (permettant de définir un ordre complet sur l'ensemble des tâches)
- Variables continues dites variables de flux f_{ijk}, où f_{ijk} représente la quantité de la ressource de type k directement transmise d'une tâche i (à la fin de son exécution) vers une tâche j (au début de son exécution).

On suppose qu'au début du projet les ressources sont toutes disponibles et stockées au niveau de la tâche fictive de début du projet. Par la suite, l'exécution des premières tâches débute par l'utilisation des ressources disponibles, et une fois terminées, elles transfèrent leurs ressources aux tâches qui les succèdent. Cette opération se répète jusqu'à ce que les dernières tâches à exécuter les transmettent à leur tour à la tâche fictive F de la fin du projet; cela se traduit par $r_{1k} = r_{Fk} = Q_k, \forall k \in K$.
Le modèle est donné par :

$$\text{Miminize } S_F \tag{10}$$

Sous les contraintes :

$$y_{ij} = 1 \qquad \forall (i,j) \in P \tag{11}$$

$$S_j - S_i - M.y_{ij} \geq d_i - M \qquad \forall i \in J \cup \{1\}, \forall j \in J \cup \{F\} \tag{12}$$

$$f_{ijk} \leq \min(r_{ik}, r_{jk}).y_{ij} \qquad \forall i \in J \cup \{1\}, \forall j \in J \cup \{F\}, \forall k \in K \tag{13}$$

$$\sum_{j \in J \cup \{F\}} f_{ijk} = r_{ik} \qquad \forall i \in J \cup \{1\}, \forall k \in K \tag{14}$$

$$\sum_{i \in J \cup \{1\}} f_{ijk} = r_{jk} \qquad \forall j \in J \cup \{F\}, \forall k \in K \tag{15}$$

$$f_{ijk} \geq 0 \qquad \forall i \in J \cup \{1\}, \forall j \in J \cup \{F\}, \forall k \in K \tag{16}$$

$$S_i \geq 0 \qquad \forall i \in J \cup \{1, F\} \tag{17}$$

$$y_{ij} \in \{0,1\} \qquad \forall i \in J \cup \{1\}, \forall j \in J \cup \{F\} \tag{18}$$

La fonction objectif donnée par l'expression (10) est la minimisation de la date de début de la dernière tâche (tâche F) du projet. Les équations (11) expriment les relations de préséance entre les tâches du projet. Les contraintes (12) permettent de lier les deux variables y_{ij} et S_i pour chaque paire de tâches i et j. Les contraintes (13) lient les variables de flux f_{ijk} aux variables séquentielles y_{ij}, via la disponibilité des ressources. Ainsi, si i précède j, alors le flux maximal de ressources envoyé de i vers j correspond au minimum de la consommation r_{ik} de la tâche prédécesseur i et de la consommation r_{jk} de la tâche successeur j, ceci, quel que soit la ressource k. Les contraintes (14) et (15) assurent la conservation des flux. Les contraintes (16) et (17) imposent que les variables de flux et les dates de début des tâches soient des variables continues. Les contraintes (18) quant à elles imposent que les variables y_{ij} soient des variables binaires.

2.3 Formulation à temps discrétisé de Pritsker et al. (1969)

La première formulation à temps discrétisé a été proposée par Pritsker et al. (1969). Dans ce type de formulation, les variables de décision sont des variables binaires indexées sur un horizon de temps discrétisé de T périodes, telle que :

$$x_{it} = \begin{cases} 1, & \text{si l'exécution de la tâche } i \text{ commence au début de la période } t \\ 0, & \text{sinon.} \end{cases}$$

Le modèle est donc donné par :

$$\text{Miminize} \sum_{t=1}^{T} t x_{Ft} \tag{19}$$

Sous les contraintes :

$$\sum_{t=1}^{T} x_{it} = 1 \qquad \forall J \cup \{1, F\} \tag{20}$$

$$\sum_{t=1}^{T} t\,(x_{jt} - x_{it}) \geq d_i \qquad \forall (i,j) \in P \tag{21}$$

$$\sum_{i \in J} r_{ik} \sum_{\tau=t-d_i+1}^{t} (x_{i\tau}) \leq Q_k, \qquad \forall t = 1, ..., T, \forall k \in K \tag{22}$$

$$x_{it} \in \{0, 1\} \qquad \forall i \in J \cup \{1, F\}, \forall t = 1, ..., T \tag{23}$$

La fonction objectif donnée par (19) représente la minimisation de la durée totale du projet. Les contraintes (20) assurent que chaque tâche doit avoir une seule date de début (la préemption est non permise). Les contraintes (21) expriment les relations de préséances entre les différentes tâches, en stipulant que si la paire de tâches (i, j) appartient à l'ensemble P, alors la date de début de la tâche j sera ultérieure ou égale à la date de fin de la tâche j. Les contraintes (22) représentent les contraintes de ressources; elles assurent que pour chaque type de ressource et à chaque période de l'horizon temporel, la somme des consommations des tâches en cours d'exécution ne dépasse pas la quantité des ressources disponibles. Les contraintes (23) assurent que les variables x_{it} prennent des valeurs binaires.

2.4 Formulation en temps discrétisé basée sur les ensembles admissibles

Cette formulation a été proposée par Mingozzi et al. (1998) elle est basée sur le principe que toute solution réalisable peut être caractérisée par une séquence, qu'on peut noter $s = (Adm_{t_1}, \ldots, Adm_{t_T})$, où Adm_{t_i} est un ensemble de tâches de tâche ($Adm_{t_i} \subseteq V$), pouvant être exécutées à la période t. Il est dit admissible, car les tâches qui le composent peuvent s'exécuter simultanément en regard des contraintes de préséance et de ressources.

Avant de présenter le modèle, on définit les deux ensembles $IndA$ et $IndA_i$ par :

- *IndA*: Ensemble des indices l de tous les ensembles admissibles de J, (toutes les tâches incluses dans un ensemble admissible doivent vérifier les contraintes de préséance et de ressource).
- $IndA_i \subseteq IndA$: Ensemble de tous les indices des ensembles admissibles incluant la tache i.

Le modèle définit les deux ensembles de variables de décision binaires suivants : x_{it} et z_{lt}. Les variables x_{it} sont definies comme dans la formulation de (Pritsker et al., 1969). Le deuxième ensemble de variable z_{lt} est définie pour tout ensemble admissible Adm, et sur toutes les périodes de l'horizon temporel T tel que :

$$z_{lt} = \begin{cases} 1, & \text{si toutes les tâches du sous ensembe admissible } l \text{ sont exécutées à la période } t \\ 0, & \text{sinon.} \end{cases}$$

Le modèle est identique au précédent, excepté les contraintes de ressource, qui sont remplacées par les contraintes suivantes :

Sous les contraintes :

$$\sum_{l \in IndA_i} \sum_{t \in I_i} z_{lt} = d_i \qquad \forall J \cup \{1, F\} \qquad (24)$$

$$\sum_{l \in IndA_i} z_{lt} \leq 1 \qquad \forall t = 1, ..., T \qquad (25)$$

$$x_{it} \geq \sum_{l \in IndA_i} z_{lt} + \sum_{l \in IndA_i} z_{l(t+1)} \qquad \forall i \in J \cup \{1, F\}, \forall t = 1, ..., T \qquad (26)$$

$$z_{lt} \in \{0, 1\} \qquad \forall l \in IndA, \forall t = 1, ..., T \qquad (27)$$

I_i définit la fenêtre de temps où la tâche i peut etre exécutée.
Les contraintes (24) garantissent que toute tâche i s'exécute pendant d_i périodes situées entre la date de début au plus tôt et au plus tard (ces dates sont définies par la méthode du chemin critique). Les contraintes (25) imposent qu'au plus un seul ensemble admissible soit en cours d'exécution à chaque période t. Les contraintes (26) imposent que la variable de décision x_{it}, prenne la valeur 1 si une tâche i est en exécution à la période t, et qu'elle n'est pas dans l'ensemble admissible en exécution à la période t-1. Les contraintes (27) quant à elles, imposent que les variables z_{lt} prennent des valeurs binaires.

Les différentes formulations mathématiques du POPRL souffrent soit de leur taille (nombre de variables ou de contraintes), soit de la faiblesse de leurs relaxations. L'inconvénient

de la formulation de Mingozzi et al. (1998) est qu'elle génère un nombre exponentiel de variables binaires; son utilisation se limite donc à des instances de très petite taille. Mais à partir de celle-ci, les auteurs ont été capables de définir des bornes inférieures d'une assez bonne qualité.

La section suivante du présent document propose un survol des différentes techniques développées dans la littérature pour définir des bornes inférieures au POPRL.

3 Bornes inférieures pour le POPRL

Les bornes inférieures sont d'une très grande utilité dans les méthodes de séparation et d'évaluation ainsi que dans les méthodes de résolution approchées. Mais le choix d'une méthode pour obtenir une borne implique un compromis entre la qualité de la borne et son temps de calcul, ce qui a incité les chercheurs à développer plusieurs techniques pour définir des bornes inférieures pour le POPRL. Ces techniques consistent à autoriser la violation de certaines contraintes du problème ou à la réduction du problème initial de façon à ce que la solution donne une borne inférieure pour la valeur optimale du problème initial. Le Tableau 1 ci-dessous classifie les différentes approches trouvées dans la littérature en deux grandes classes (voir Klein (1999) et Demeulemeester and Herroelen (2006). Celles dites directes, elles peuvent être simples ou complexes, et celles dites indirectes.

Dans les trois sous sections qui suivent, nous présenterons respectivement des bornes inférieures issues d'une technique directe simple, d'une technique directe complexe et d'une technique indirecte.

3.1 Borne inférieure du chemin critique

La borne inférieure la plus simple et la plus utilisée est la borne du chemin critique. Elle consiste à relaxer les contraintes de ressources du POPRL. La résolution du problème obtenu se résume tout simplement à chercher le chemin le plus long dans un graphe $G = (J \cup \{1, F\}, P)$. Ce chemin porte l'appellation de chemin critique, ainsi sa longueur définie une borne inférieure sur la solution optimale du POPRL.

Différentes généralisations sur le chemin critique ont permis d'améliorer cette borne. Ainsi, Stinson et al. (1978) proposent la borne définie par la fonction (28) ci-dessous:

$$CPM + \max_i(d_i - e_i) \tag{28}$$

où CPM est la longueur du chemin critique et e_i est la longueur d'une partie de la tâche i qui peut être exécutée en parallèle avec le chemin critique sans violer la contrainte de disponibilité des ressources.

3.2 Borne inférieure de Mingozzi et al (1998)

Mingozzi et al. (1998) ont proposé cinq bornes inférieures pour le POPRL. Celles-ci dominent la borne du chemin critique, et elles sont plus serrées que la borne qui a été définie par Stinson et al. (1978). Elles ont toutes été obtenues à partir de la formulation du POPRL basée sur les ensembles admissibles, et par l'utilisation de différents types de relaxations (relaxation linéaire, relaxation des contraintes de préséance, ...etc.). L'idée de base est de trouver un sous ensemble de tâches dont aucune paire ne peut être exécutée en parallèle. La somme des durées de ces tâches constitue une borne inférieure sur la durée du projet. Les résultats expérimentaux démontrent que les bornes obtenues présentent un bon compromis qualité/temps de calcul.

3.3 Borne d'amélioration destructive de Klein and Scholl (1999)

Dans l'approche dite d'amélioration destructive, la borne inférieure est la plus petite valeur de T pour laquelle on ne peut plus prouver que le problème est réalisable.

4 Méthode exactes de résolution du POPRL

Les problèmes d'ordonnancement de projet à ressources limitées sont des problèmes NP-durs (voir Blazewicz et al. (1983)). Les méthodes exactes de résolution de ce problème peuvent se définir en trois classes (voir Kolisch (2000)) :

1. Méthodes basées sur la programmation en nombres entiers;
2. Méthodes basées sur la programmation dynamique;
3. Méthodes de recherche arborescente.

Bornes directes
Simples
1. Borne basée sur le chemin critique - le chemin critique : CPM - séquence critique : Stinson et al. (1978) - séquences étendues : Demeulemeester and Herroelen (1992) 2. Borne basée sur les ressources - la quantité de travail maximale - la quantité de travail étendue 3. Borne définie par les n-machine : Carlier and Latapie (1991) 4. Borne définit par la résolution d'un problème du "node packing" de base Mingozzi et al. (1998) 5. Borne définie par une version améliorée du problème "node packing" Mingozzi et al. (1998) 6. Borne définie par une version des n-machine : Mingozzi et al. (1998) 7. Borne basée sur les sous projet : Klein and Scholl (1999)
Complexes
1. Borne définie par la relaxation linéaire Christofides et al. (1987) 2. Bornes définie par relaxation Lagrangienne Christofides et al. (1987) 3. Borne définie par une approche de "set covering" :Mingozzi et al. (1998)
Bornes indirectes
Klein and Scholl (1999)

Tableau 1: Classification des techniques pour trouver des bornes inférieures

Patterson (1984) a souligné dans une étude comparative de ces trois méthodes, que les méthodes de recherche arborescente sont les méthodes les plus performantes et les plus efficaces. Le principe général de ces méthodes est de construire un arbre de recherche afin d'explorer implicitement l'espace de solutions, ce qui revient à partitionner le problème en sous problèmes pour pouvoir ensuite éliminer ceux qui sont moins intéressants en se basant sur soit des bornes inférieures soit des règles de dominance. L'opération de partitionnement s'appelle la séparation ou le branchement et l'élimination est la résultante d'une évaluation.

Dans la littérature, plusieurs techniques d'évaluation et de séparation ont été proposées pour le POPRL. Elles se distinguent principalement par le choix du schéma de branchement (la séparation de l'espace de recherche) et le choix de la méthode d'évaluation (bornes inférieures, règle de dominance). On distingue principalement les trois types de branchement suivants :

- Branchement d'une seule tâche par nœud;
- Branchement d'un sous-ensemble de tâches par nœud;
- Branchement basé sur la fixation des paires de disjonctions et des tâches pouvant être exécutées en parallèle.

Les trois sections qui suivent présentent les grandes lignes dans chaque type de branchement. Il s'agit respectivement de l'algorithme de Patterson et al. (1989) basé sur le branchement d'une seule tâche par nœud, de l'algorithme de Demeulemeester and Herroelen (1992) basé sur un ensemble de tâches par nœud, et enfin, l'algorithme basé sur les schémas d'ordonnancement de Brucker et al. (1998).

4.1 Algorithme de Patterson et al. (1989)

Cet algorithme a été proposé par Patterson et al. (1989). Il a ensuite été révisé et généralisé dans d'autres travaux pour traiter d'autres variantes du problème. L'idée de base de cet algorithme de séparation et d'évaluation est l'énumération (implicite) de toutes les séquences réalisables du problème. Une séquence, appelée aussi liste, est dite réalisable si l'ordre attribué aux tâches respecte les contraintes de préséance, c'est-à-dire, qu'aucune tâche n'est listée avant l'un de ses prédécesseurs.

À l'étape initiale de l'algorithme, la tâche fictive 1 du début du projet est planifiée à une date de début égale à zéro. Ensuite à chaque niveau dans l'arbre, l'algorithme détermine

l'ensemble des tâches déjà planifiées et l'ensemble des tâches éligibles (ceux dont les prédécesseurs ont été déjà planifiés). Une tâche éligible est ensuite sélectionnée, et sa date de début au plus tôt est calculée en tenant compte des contraintes de ressources et de préséance. L'algorithme passe ensuite au niveau suivant. Si la dernière tâche fictive du projet F est planifiée, alors un ordonnancement réalisable est complété. Dans ce cas, un retour vers les niveaux précédents est prévu et la prochaine tâche éligible non encore testée est choisie. Si toutes les tâches éligibles ont été testées, l'algorithme remonte encore d'un niveau dans l'arbre.

4.2 Algorithme de Demeulemeester and Herroelen (1992)

Cet algorithme a été initié par Christofides et al. (1987) et il a été par la suite révisé et amélioré par Demeulemeester and Herroelen (1992). Il est basé sur la notion de « l'alternative de retard minimal (Minimal Delaying Alternative) » où un MDA est lié au concept d'ensemble réalisable maximal (ensembles maximal de tâches qui peuvent être exécutées simultanément) et au concept d'ensemble interdit IN (ensembles des tâches qui ne sont pas reliées par des relations de préséance mais ne pouvant pas être exécutées simultanément sans provoquer un conflit de ressource). Un MDA est un sous ensemble réalisable maximal d'un ensemble interdit.

À l'étape initiale de l'algorithme, la tâche fictive 1 du début du projet est planifiée à une date de début égale à zéro. Ensuite chaque niveau de l'arbre est associé à un instant t_g (dit instant de décision) auquel toutes les tâches éligibles (dont tous leurs prédécesseurs sont déjà planifiés et se terminent tous avant t_g) sont planifiées si elles ne provoquent pas un conflit de ressources, et ainsi un seul nœud est rajouté à l'arbre. Dans le cas contraire, l'algorithme énumère tous les MDA possibles (une branche pour chaque MDA).

Demeulemeester and Herroelen (1992) ont utilisé cet algorithme pour résoudre les instances de Patterson (1984). Les résultats expérimentaux obtenus restent compétitifs jusqu'à ce jour.

4.3 Algorithme de Brucker et al. (1998)

Ce type d'algorithme a été développé par Brucker et al. (1998). Il est basé sur une représentation assez particulière, puisqu'à chaque nœud de l'arbre correspond un schéma d'ordonnancement, noté (C, D, N, L), où C est l'ensemble des paires de tâches en conjonction (si $(i, j) \in C$, alors i précède j), D est l'ensemble des paires en disjonction

$((i,j) \in D$, donc la somme des consommations des tâches i et j excède la quantité totale disponible pour au moins une ressource, mais l'ordre d'exécution des deux tâches est encore inconnu), N inclut toutes les tâches qui peuvent s'exécuter en parallèle pour au moins une période de l'horizon temporel, et L est l'ensemble des tâches flexibles.

L'algorithme est initialisé par un schéma initial $(C_0, D_0, \emptyset, L)$ tel que C_0 est défini par toutes les contraintes de préséance du départ du problème, D_0 est l'ensemble des paires de tâches (i,j) tel que $(r_{ik} + r_{jk} \geq Q_k)$ pour au moins une ressource $\forall k \in K$, et L groupe toutes les paires de tâches qui ne sont pas liées par une relation de disjonction ou de conjonction.

L'objectif du branchement est de remplacer les relations de flexibilité par des relations de conjonction ou des relations parallèles. Et à partir du schéma exprimé par $(C, D, N, \emptyset)$, il est possible de trouver en un temps polynomial le meilleur ordonnancement correspondant définit par $(C, \emptyset, N, \emptyset)$. L'algorithme effectue ensuite des retours en arrière pour améliorer la solution courante.

5 Méthodes heuristiques

Les méthodes heuristiques pour la résolution des problèmes combinatoires de type NP-dur sont d'une grande utilité puisqu'elles permettent de donner de bonnes solutions dans des temps d'exécution acceptables. C'est pour cette raison qu'elles ont été d'une grande popularité dans la résolution des POPRL. Elles ont été classées par Hartmann (2012) comme suit :

1. Les heuristiques constructives basées sur des règles de priorité;
2. Les méthodes de recherche dans le voisinage;
3. Les techniques d'énumération tronquée;
4. Les approches basées sur les arcs disjonctifs.

Ces méthodes sont abordées dans chacune des sections suivantes.

5.1 Heuristiques constructives basées sur des règles de priorité

Ce type de méthodes a été d'une grande popularité dans le domaine de l'ordonnancement de projet. Elles sont des méthodes intuitives, faciles à mettre en œuvre et très rapides.

Elles sont constructives parce que la solution est obtenue d'une façon progressive, en prenant une décision à la fois. À chaque étape la tâche éligible ayant la plus grande priorité est planifiée; si deux tâches ou plus sont à égalité, une deuxième règle de priorité doit être utilisée.

Ces heuristiques se divisent en deux groupes : les heuristiques sérielles et les heuristiques parallèles. Pour les heuristiques sérielles, les indices de priorité sont fixés avant de commencer à planifier les tâches tandis que pour les heuristiques parallèles, l'indice de priorité est recalculé après chaque décision. Les pseudo codes de ces deux types d'heuristiques sont donnés dans la Figure 1 et 2 respectivement (Kolisch (1996)).

Établir un ordre de priorité de toutes les tâches à planifier qui respecte les relations de préséance

Tant qu'il reste encore des tâches non planifiées :

1. Planifier chaque tâche selon son ordre de priorité à la date la plus proche de sa date de début au plus tôt et qui respecte la disponibilité des ressources (vérifier la disponibilité des ressources à chaque périodes de sa durée d'exécution)
2. Mettre à jour la disponibilité des ressources en conséquence et mettre à jour la liste des tâches non planifiées.

Figure 1: Pseudo code des heuristiques dites sérielles

On distingue aussi les heuristiques à passe unique et celles à passes multiples. Les heuristiques à passes multiples construisent à chaque passe une nouvelle solution soit en utilisant une autre règle de priorité soit en inversant le sens d'exécution de l'heuristique tel que décrit dans les trois paragraphes suivants.

1. **Heuristiques de construction « en avant »** : dans ce type de construction, les tâches sont planifiées après leurs prédécesseurs et on détermine leur dates de début en fonction des dates de fin de ces prédécesseurs. On commence donc par l'ordonnancement des tâches sans prédécesseurs en leur accordant la date de début au plus tôt du projet et on continu la procédure jusqu'à ce qu'on atteigne les tâches sans successeurs.

Tant qu'il reste encore des tâches non planifiées :

1. Avancez à la période la plus proche où il y a des ressources encore disponibles;
2. Etablir la liste des tâches qui sont éligibles à cette période (c'est à dire ceux pour lesquels toutes les ressources encore disponibles sont suffisantes pour pouvoir les exécuter et tous leurs prédécesseurs immédiats sont déjà planifiées et déjà terminées à cette période.
3. Établir un ordre de priorité des tâches éligibles.
4. Si la liste est vide revenir à l'étape 1
5. Tant qu'il reste encore des ressources disponibles, planifier les tâches éligibles selon leurs ordre de priorité.
6. Mettre à jour la disponibilité des ressources en conséquence et mettre à jour la liste des tâches éligibles et les tâches déjà planifiées à la période de temps considérée.
7. Retournez à l'étape 4.

Figure 2: Pseudocode des heuristiques dites parallèles

2. **Heuristiques de construction « en reculant »** : dans ce cas, on commence par planifier les tâches sans successeurs en leur donnant une date de fin quelconque. Leurs dates de début sont ensuite calculées et utilisées comme date de fin pour leurs prédécesseurs. À la fin, toutes les dates sont décalées de façon à commencer le projet à sa date de début au plus tôt.

3. **Heuristiques de construction bidirectionnelle** : les deux techniques précédentes sont alternativement exécutées pour donner deux ordonnancement (un ordonnancement en avant et un en reculant). La meilleure des deux solutions est ensuite retenue.

   Dans la littérature on constate qu'il existe une grande panoplie de règles (voir Boctor (1990),Olaguibel and Goerlich (1993) et Kolisch (1996) pour plus de détail). Le Tableau 2 ci-dessous résume quelques unes d'entre elles selon la classification qui a été proposée par Klein (1999) :

   a. Des règles basées sur les caractéristiques des tâches,

   b. Des règles basées sur le chemin critique,

c. Des règles basées sur les ressources,

d. Des règles hybrides.

Classe	Acronyme	Signification
Des règles basées sur les caractéristiques des tâches	SPT	Plus petite durée
	MIS	Le plus grand nombre de successeurs immédiats
	MTS	Le plus grand nombre de successeurs
	RPW	La grande somme des durées de la tâche avec tous ses successeurs immédiats
	RPW*	La grande somme des durées de la tâche avec tous ses successeurs
Des règles basées sur le chemin critique	EST	Plus petite date de début au plus tôt
	EFT	Plus petite date de fin au plus tôt
	ESTD	Plus petite date de début au plus tôt (dynamique)
	LST	Plus petite date de début au plus tard
	LFT	Plus petite date de début au plus tard
	MSL	Plus petite marge totale
	MSLD	Plus petite marge totale (dynamique)
Des règles basées sur les ressources	WRUP	Le poids le plus élevé de l'utilisation des ressources et les règles de préséance

Tableau 2: Classification des règles de priorité pour le POPRL

5.2 Méthodes de recherche dans le voisinage

Ces méthodes peuvent donner de meilleurs résultats que les heuristiques basées sur des règles de priorité puisque à partir d'une solution de départ (donnée par une heuristique constructive par exemple), il est possible d'améliorer cette solution en explorant ses voisinages. Les deux ingrédients essentiels pour définir un voisinage sont le schéma de représentation (le codage de la solution) et les opérateurs de voisinage.

1. **Schéma de de représentation des solutions** : En faisant un survol de la littérature sur les POPRL, on remarque que cinq différents schémas ont été utilisés pour représenter une solution. On trouve :

   a. **Représentation par une liste ou une séquence** : L'ordonnancement (solution) peut être ensuite obtenu en attribuant à chaque tâche la date de début au plus tôt en se basant sur les relations de préséance et la disponibilité des ressources, voir Boctor (1996b).

b. **Représentation par une liste ordonnée de règles de priorité** : pour ce type de représentation, la séquence est exprimée par un vecteur, chaque composante du vecteur étant une règle de priorité. L'ordonnancement peut être ensuite déduit en appliquant ces règles dans l'ordre préétabli, voir Artigues et al. (2003).

c. **Représentation par un vecteur de clés**: pour ce type de représentation, à chaque tâche est assigné un nombre réel défini sur un intervalle quelconque (par exemple l'intervalle [0 1[). En classant les tâches selon l'ordre croissant ou décroissant, on obtient une représentation en liste et à partir de celle-ci nous pouvons déduire un ordonnancement, voir Gonçalves et al. (2011).

d. **Représentation par vecteur décalé** : la quatrième représentation a été proposée par Sampson and Weiss (1993). Chaque solution est représentée par un vecteur, où chaque composante du vecteur est un entier non négatif, indiquant le nombre de périodes d'exécution qu'une tâche donnée doit être décalée (retardée) par rapport à la date au plus tôt. Un ordonnancement est ensuite obtenu en utilisant les dates ainsi décalées. Évidemment, cette représentation permet de visiter des voisins non réalisables puisque ces dates peuvent mener à des violations des limites de ressources. Par conséquent, une pénalité doit être appliquée à la valeur de la fonction objectif des solutions non réalisables.

e. **Représentation par schéma d'ordonnancement** : Cette représentation a été utilisée par Brucker et al. (1998). Elle se base sur le principe d'exprimer la relation qui existe entre chaque paire de tâches du projet, en fonction de quatre types de relations : Conjonction, Disjonction, Parallèle ou Flexible. Ces relations définissent respectivement les quatre ensembles disjoints (C, D, P, L). L'ordonnancement est ensuite obtenue par l'heuristique de Baar et al. (1999).

2. **Opérateurs de recherche dans le voisinage** : ce sont des opérations élémentaires appliquées sur une ou deux solutions pour aboutir à une ou plusieurs solutions voisines. Parmi les opérateurs de voisinage, on trouve :

   a. **Opérateurs pouvant être appliqués sur une seule solution** : Partant d'une solution dite courante, une solution voisine peut être obtenue par un :

      i. Opérateur de déplacement où une tâche est déplacé à une nouvelle position dans la liste définissant la solution

ii. Opérateur de permutations où deux ou plusieurs tâches échangent leur position,

iii. Opérateur de décalage où une partie de la liste est déplacée (décalée).

b. **Opérateurs applicables sur deux ou plusieurs solutions** À partir de deux ou plusieurs solutions, d'autres solutions voisines peuvent être obtenues par :

i. Un croisement des solutions (croisement à une seule position, croisement à plusieurs positions, croisement uniforme),

ii. Une combinaison (linéaire ou non) de ces solutions.

Plusieurs approches de recherche dans le voisinage ont été proposées dans la littérature. Le Tableau 3 présente une classification de quelques-unes de ces méthodes en se limitant aux trois heuristiques de recherche dans le voisinage les plus utilisées (recherche tabou, algorithme génétique et l'algorithme de recuit simulé). Pour chacun de ces travaux, le tableau donne la méthode employée, le code de représentation de la solution, l'heuristique utilisée pour déduire un ordonnancement du code de présentation, et les opérateurs de recherche dans le voisinage utilisé.

Kolisch and Hartmann (2006) ont comparé plusieurs de ces méthodes, notamment, des méthodes de recherche tabou, des algorithmes génétiques, des algorithmes de recuit simulé ainsi que plusieurs méthodes hybrides. Leurs résultats expérimentaux démontrent que les méthodes hybrides sont les plus prometteuses, comme celle de Kochetov and Stolyar (2003), où l'algorithme génétique est hybridé avec une méthode de recherche tabou.

Des méthodes aussi efficaces que celle de Kochetov and Stolyar (2003) ont vu le jour ultérieurement. Debels et al. (2006) ont utilisé la nouvelle technique dite de recherche dispersée (Scatter Search). Chen (2011) et Koulinas et al. (2014) ont proposé la méthode dite Particle swam optimisation. Valls et al. (2008), Mendes et al. (2009), Montoya-Torres et al. (2010) et Zamani (2013) quant à eux ont utilisé des algorithmes génétiques.

Le Tableau 4 propose une classification des résultats obtenus par ces méthodes quand elles ont été testées sur les trois ensembles de problème test de Kolisch and Sprecher (1997) : les ensembles appelés J30 (avec 30 tâches), J60 (60 tâches) et J120 (120 tâches). Les résultats sont obtenus en limitant à 5000 le nombre de solutions générées.

	Auteurs	Code de représentation de la solution	Méthode de construction de la solution	Opérateurs de recherche dans le voisinage
Recherche Tabou	Lee and Kim (1996)	Liste aléatoire	parallèle	Permutation
	Baar et al. (1999)	Liste de priorité	sériel	décalage
	Thomas and Salhi (1998)	Schéma d'ordonnancement	Autre	
	Klein and Scholl (1999)	Ordonnancement	-	trois mouvements
	Artigues et al. (2003)	Liste de priorité	sériel	décalage
Algorithme Génétique	Lee and Kim (1996)	Clé aléatoire	parallèle	croisement à 1 points
	Hartmann (1998)	Règle de priorité	sériel	croisement à 2 points
		Clé aléatoire	sériel	croisement à 2 points
	Kohlmorgen et al. (1999)	Clé aléatoire	sériel	croisement à 2 points
	Hartmann (2002)	Liste de priorité	sériel/parallèle	croisement à 2 points
	Hindi et al. (2002)	Liste de priorité	sériel	croisement à 2 points
	Toklu (2002)	Ordonnancement	-	-
	Coelho and Valadares (2002)	Liste de priorité	sériel	croisement à 2 points
	Valls et al. (2008)	Liste de priorité	sériel	croisement à 2 points
	Mendes et al. (2009)	Clé aléatoire	Procédure adaptée	croisement à 2 points
	Montoya-Torres et al. (2010)	Liste de priorité	sériel	croisement à 2 points
	Zamani (2013)	Liste de priorité	sériel/parallèle	croisement à 2 points modifié
Recuit simulé	Sampson and Weiss (1993)	Vecteur de décalage	-	-
	Boctor (1996b)	Liste de priorité	sériel	Permutation
	Lee and Kim (1996)	Clé aléatoire	parallèle	Permutation
	Bouleimen and Lecocq (2003)	Liste de priorité	sériel	Permutation
	Valls et al. (2004)	Clé aléatoire	sériel/parallèle	Permutation

Tableau 3: Approches de recherche dans le voisinage proposées dans la littérature pour le POPRL.

Auteurs	Méthode de recherche dans le voisinage	Déviation moyenne par rapport		
		l'optimum	à longueur du chemin critique	
		J30	J60	J120
Kochetov and Stolyar (2003)	Hybride	0.04	11.17	32.06
Debels et al. (2006)	Scatter Search	0.11	11.10	33.10
Valls et al. (2008)	Algorithme Génétique	0.06	11.10	**32.54**
Mendes et al. (2009)	Algorithme Génétique	**0.02**	11.04	33.03
Chen (2011)	Particle swam optimisation	0.14	11.43	33.88
Zamani (2013)	Algorithme Génétique	0.04	**10.94**	32.89
Koulinas et al. (2014)	Particle swam optimisation	0.04	11.13	32.59

Tableau 4: Comparaison des résultats de quelques travaux récents

5.3 Techniques d'énumération tronquée

Pour obtenir des solutions de bonne qualité pour le POPRL, en des temps de calcul raisonnables, quelques auteurs ont opté pour des méthodes d'énumération telles que la méthode de séparation et évaluation, mais en appliquant une règle qui permet d'arrêter le déroulement de la méthode après un temps prédéterminé ou en limitant le nombre de nœuds visités dans l'arbre d'énumération (voir Pollack-Johnson (1995), Demeulemeester and Herroelen (1997), Sprecher (2000), et Demeulemeester and Herroelen (2006), pour plus de détails).

5.4 Approches basées sur l'ajout des arcs disjonctifs

Cette approche a trouvé son application dans les travaux de Alvarez-Valdes and Tamarit (1989) et Bell and Han (1991). L'idée de base de cette approche est l'ajout de nouvelles relations de préséance au problème original, dans l'objectif de rendre le calendrier au plus tôt réalisable avec les ressources disponibles.

6 Généralisation du POPRL

Le POPRL tel qu'il a été défini précédemment est une simplification des situations réelles, ce qui a incité les chercheurs ces dernières années à proposer plusieurs généralisations du problème standard. Hartmann and Briskorn (2010) ont proposé une revue bibliographique dans laquelle ils ont rapporté quelque cent quatre-vingt-dix-huit références traitant des variantes ainsi que des généralisations du problème. Dans le

Tableau 5, on propose un bref survol de quelques-unes de ces variantes et généralisation divisées en trois groupes:

1. Variantes avec des objectifs alternatifs,
2. Généralisations avec des contraintes de préséance généralisées,
3. Variantes avec de nouvelles hypothèses.

Dans les sections qui suivent, la littérature relative à trois de ces généralisations va être présentée. La section 7, présente la littérature traitant le problème où les tâches peuvent avoir plusieurs modes d'exécution. Cette version plus réaliste du POPRL a fait l'objet d'un nombre improtant de publications. La section 8 présente la littérature relative au problème avec temps de transfert des ressources entre les sites d'execution des tâches. On constate qu'il existe très peu de travaux consacrée a ce problème. La section 9 est consacrée au problème d'ordonnancement de projet dans l'objectif de minimiser le coût d'exécution du projet. Nous verons que la plupart des publications ne prennent pas en considération tous les éléments du coût.

7 Problème d'ordonnamcement de projet à ressources limitées avec plusieurs modes d'exécution-POPRL/PME

Le problème d'ordonnancement de projet sous contrainte de ressources limitées où les tâches ont plusieurs modes d'exécution sera noté POPRL/PME. Ce problème est une généralisation du problème standard permettant que chaque tâche ait un nombre limité de modes d'exécution où chaque mode est caractérisé par les quantités des ressources renouvelables et non renouvelables requises et la durée d'exécution que l'on peut obtenir si ces quantités de ressources sont utilisées pour exécuter la tâche. Dans un POPRL/PME, il s'agit non seulement de trouver les dates de début ou de fin des tâches, mais aussi de choisir le mode d'exécution à utiliser pour chaque tâche avec l'objectif de minimiser la durée du projet. Le POPRL/PME a été d'une grande popularité ces dernières années, une revue de littérature est proposé par Węglarz et al. (2011).

Dans la section suivante, on définit le problème d'une façon un peu plus formelle, la deuxième et la troisième sous-section présenteront un survol des approches de résolution exactes et heuristiques respectivement.

Variantes avec d'autres objectifs	Objectif basé sur le temps	Nudtasomboon and Randhawa (1997); Kolisch (2000); Viana and de Sousa (2000)
	Objectif basé sur la robustesse de la cédule	Icmeli-Tukel and Rom (1997); Al-Fawzan and Haouari (2005) Abbasi et al. (2006);Kobylański and Kuchta (2007); Chtourou and Haouari (2008)
	Objectif pour le réordonnancement	El Sakkout and Wallace (2000); Vanhoucke et al. (2001) Calhoun et al. (2002); Van de Vonder et al. (2007)
	Objectif basé sur les coûts	Maniezzo and Mingozzi (1999); Möhring et al. (2001) Khoshjahan et al. (2013);Gomes et al. (2014)
	Maximiser la valeur actuelle nette	Vanhoucke et al. (2001); Mika et al. (2005); Kim et al. (2005); Padman and Zhu (2006); Vanhoucke (2010) Leyman and Vanhoucke (2015);Leyman and Vanhoucke (2016)
	Objectifs multiples	Nudtasomboon and Randhawa (1997); Hapke et al. (1998) Al-Fawzan and Haouari (2005); Doerner et al. (2008) Xiao et al. (2016)
Contraintes de temps généralisées	Décalage minimal et maximal entre la fin de la tâche et le début d'un successeur	Bartusch et al. (1988); Neumann and Zhan (1995) Herroelen et al. (1998); De Reyck et al. (1998) Kreter et al. (2016)
Généralisations avec des hypothèses différentes	Preemption permise	Franck et al. (2001); Buddhakulsomsiri and Kim (2006) Damay et al. (2007); Ballestín et al. (2008)
	Temps de préparation avant l'exécution de tâches	Van de Vonder et al. (2007); Mika et al. (2008)
	Tâches ayant plusieurs modes d'exécution	Talbot (1982); (Boctor, 1993, 1996b,a); Mori and Tseng (1997) Boctor and d'Avignon (2005); Zhu et al. (2006); Zhu et al. (2006) Vanhoucke et al. (2008); Van Peteghem and Vanhoucke (2010) Coelho and Vanhoucke (2011); Van Peteghem and Vanhoucke (2014) Messelis and De Causmaecker (2014)
	Temps de transfert des ressources	(Krüger and Scholl, 2009; Krüger, 2009) Poppenborg and Knust (2014)

Tableau 5: Classification de quelques variantes et généralisations du POPRL.

7.1 Modélisation

Pour modéliser un POPRL/PME, Talbot (1982) a proposé une généralisation de la formulation à temps discrétisée de Pritsker et al. (1969). Il définit les variables de décision comme suit :

$$x_{imt} = \begin{cases} 1, & \text{si l'exécution de la tâche } i \text{ commence au début de la période } t \\ & \text{en utilisant le mode } m \\ 0, & \text{sinon.} \end{cases}$$

$$\text{Minimiser} \sum_{t \in I_i} t x_{Ft} \tag{29}$$

Sous les contraintes :

$$\sum_{m \in M_i} \sum_{t \in I_i} x_{imt} = 1 \qquad \forall i \in J \cup \{1, F\} \tag{30}$$

$$\sum_{m' \in M_j} \sum_{t \in I_j} t\, x_{jm't} - \sum_{m \in M_i} \sum_{t \in I_i} (t + d_{im})\, x_{imt} \geq 0 \qquad \forall (i,j) \in P \tag{31}$$

$$\sum_{i \in J \cup \{1,F\}} \sum_{m \in M_i} r_{imk} \sum_{\tau = max(t - d_i, EST_i)}^{min(t-1, LST_i)} x_{im\tau} \leq Q_k \qquad t = 1, ..., T \in K \forall k \in K \tag{32}$$

$$\sum_{i \in J \cup \{1,F\}} \sum_{m \in M_i} \sum_{t = I_i} n_{imw} x_{imt} \leq N_w \qquad \forall w \in W \tag{33}$$

$$x_{imt} \in \{0, 1\} \qquad \forall i \in J \cup \{1, F\}, \forall m \in M_i \forall t \in I_i \tag{34}$$

L'objectif est la minimisation de la durée du projet, il est donné par l'expression (29). Les contraintes (30) imposent qu'à chaque tâche soit attribué un seul mode d'exécution et une seule date de début (préemption non permise). Les contraintes de préséances sont exprimées par (31), quant aux contraintes de ressources renouvelables et non renouvelables elles sont données respectivement par les contraintes (32) et (33). Les contraintes (34) assurent que toutes les variables de décision du modèle prennent des valeurs binaires.

Une nouvelle formulation du POPRL/PME a été proposée par Maniezzo and Mingozzi (1999). Les auteurs ont proposé une généralisation de la formulation basée sur les ensembles admissibles de Mingozzi et al. (1998).

7.2 Méthodes de résolution exactes

Le PORPL/PME est un problème de complexité NP-dur. Les approches de résolution exactes qui ont été développées pour la résolution de ce type de problème sont des méthodes d'énumération implicite :

- Algorithmes basés sur le branchement d'une seule tâche par nœud : ils ont été proposés par : Patterson et al. (1989), Sprecher et al. (1997) et par Hartmann and Drexl (1998).
- Algorithme basé sur les alternatives de retard : il a été proposé par Sprecher et al. (1997).
- Algorithme basé sur les alternatives d'extension : il a été proposé par Sprecher et al. (1997), inspiré de l'algorithme de Demeulemeester and Herroelen (1992).

Hartmann and Drexl (1998) ont mené une étude comparative entre ces trois types d'algorithmes pour la résolution des instances de 10,12, 14 et 16 tâches de Kolisch and Sprecher (1997). Ils ont conclu que :

- Algorithme basé sur l'arbre de préséance proposé par Sprecher et al. (1997) est le plus facile à implémenter et le mieux adapté pour les grandes instances.
- Résolution exacte des problèmes de 20 tâches et plus par des méthodes exactes dans des temps acceptables reste toujours une tâche très ardue.

Une nouvelle méthode a été proposée par Zhu et al. (2006). C'est une méthode qui combine la génération d'inégalités valides et la méthode de séparation et coupe (ou Branch and Cut). Cette méthode a été testée sur les problèmes de la librairie PSLIB de Kolisch and Sprecher (1997). L'algorithme, contrairement au précédent, a réussi de trouver la solution optimale de tous les problèmes test de 20 tâches. Pour les problèmes à 30 tâches, il a trouvé l'optimum pour 506 problèmes et a trouvé la meilleure solution connue pour 529 problèmes parmi les 552 de ces ensembles de problèmes tests.

7.3 Méthodes heuristiques de résolution

Les heuristiques proposées dans la littérature pour résoudre le POPRL/PME se divisent en deux groupes : celles conçues pour la version avec des ressources renouvelables

uniquement et celles qui traitent les problèmes avec ressources renouvelables et non renouvelables. Parmi les heuristiques pour la version avec des ressources renouvelables, on peut citer (Boctor, 1993, 1996a,b; Boctor and d'Avignon, 2005). Le Tableau 6 présente une classification de ces travaux selon l'approche de résolution adoptée.

Approche de résolution	Auteurs	Instances de test	Nombre tâche	Nombre de modes d'exécution	Nombre de types de ressources
Constructive	Boctor (1993)	Boctor	50/00	1 à 4	1, 2 ou 4
	Boctor (1996a)	Boctor	50/100	1 à 4	1, 2 ou 4
	Lova et al. (2006)	Boctor	50/100	1 à 4	1, 2 ou 4
Algorithme génétique	Mori and Tseng (1997)	Autuers	20 : 70	2 à 4	4
Recherche tabou	Boctor and d'Avignon (2005)	Boctor	50/100	1 à 4	1, 2 ou 4
Recuit simulé	Boctor (1996b)	Boctor	50/100	1 à 4	1, 2 ou 4

Tableau 6: Heuristiques de résolution du POPRL/PME avec ressources renouvelables.

Les heuristiques développées pour résoudre le POPRL/PME avec des ressources renouvelables et non renouvelables se devisent en trois groupes (voir Tableau 7):

- Des heuristiques constructives basées sur les règles de priorité
- Des méthodes de recherche dans le voisinage
- Des méthodes d'évaluation et de séparation tronquées.

Van Peteghem and Vanhoucke (2014) ont effectué une étude comparative entre les meilleures heuristiques développées dans la littérature pour la résolution des POPRL/PME en utilisant les instances de PSPLIB de (Kolisch and Sprecher, 1997) avec 10, 12, 14, 16, 18, 20 et 30 tâches; les problèmes-test de Boctor (1993) à 50 et 100 tâches et d'autre problèmes-test proposés par les auteurs, només MMLIB50, MMLIB100 et MMLIB+ incluant 50, 100 et entre 50 et 100 tâches respectivement. Les resultats obenus sont résumés dans les deux tableaux Tableau 8 et Tableau 9. Où les déviations pour les instances de Kolisch et al. (1996) de 10 à 20 tâches sont mesurées par rapport à l'optimum. Pour les instances de grande taille, c'est à dire les instances J30 de Kolisch et al. (1996), et celles de 50 et 100 tâches de Boctor (1996b), et celles deVan Peteghem and Vanhoucke (2014), les déviations sont mesurés par rapport à la longueur du chemin critique.

Approche	Auteurs	Instances	Nombre de de tâche	Nombre de mode tâche	Ressources renouvelables/non renouvelables
Recherche locale	Kolisch and Drexl (1997)	PSPLIB	30	3	2/2
Algorithme génétique	Hartmann (2001)	PSPLIB	30	3	2/2
	Alcaraz et al. (2003)	PSPLIB Boctor	30 100	3 1 à 4	2/2 1 à 4/0
	Lova et al. (2009)	PSPLIB Boctor	30 100	3 1 à 4	2/2 1 à 4/0
	Tseng and Chen (2009)	PSPLIB	30	3	2/2
	Van Peteghem and Vanhoucke (2010)	PSPLIB Boctor	30 100	3 1 à 4	2/2 1 à 4/0
Recherche tabou	Nonobe and Ibaraki (2002)	PSPLIB	30	3	2/2
Recuit simulé	Józefowska et al. (2001)	PSPLIB	30	3	2/2
	Bouleimen and Lecocq (2003)	PSPLIB Boctor	30 100	3 1 à 4	2/2 1 à 4/0
Colonies de fourmis	Chyu et al. (2005)	PSPLIB	30	3	2/2
	Chiang et al. (2008)	PSPLIB	30	3	2/2
Algorithme évolutionnaire	Elloumi et al. (2006)	PSPLIB	30	3	2/2
	Damak et al. (2009)	PSPLIB	30	3	2/2
PSO	Zhang et al. (2006)	PSPLIB	30	3	2/2
	Jarboui et al. (2008)	PSPLIB	30	3	2/2
PLA	Jedrzejowicz and Ratajczak (2006)	PSPLIB	30	3	2/2
Décomposition de Benders	Maniezzo and Mingozzi (1999)	PSPLIB	30	3	2/2
	Boschetti and Maniezzo (2009)	PSPLIB	30	3	2/2
recherche dispersée	Van Peteghem and Vanhoucke (2011)	PSPLIB	30	3	2/2

Tableau 7: Heuristiques de résolution du POPRL/PME avec ressources non renouvelables.

Récemment, Messelis and De Causmaecker (2014) ont exploré une nouvelle alternative intéressante, c'est à dire la possibilité de construire un outil de sélection automatique d'algorithme pour le POPRL/PME basé sur les caractéristiques des problèmes tests. Leurs idée repose sur le concept de combiner différents algorithmes dans un super-algorithme afin d'asuurer une meilleure performance. Cette approche a prouvé son efficacité pour les problèmes-tests MMLIB50, MMLIB100 et MMLIB+ de Van Peteghem and Vanhoucke (2014)

Auteurs	Instances						
	J10	J12	J14	J16	J18	J20	J30
Van Peteghem and Vanhoucke (2011)	0.00	0.02	0.08	0.15	0.23	0.32	13.66
Van Peteghem and Vanhoucke (2010)	0.01	0.09	0.22	0.32	0.42	0.57	13.75
Lova et al. (2009)	0.06	0.17	0.32	0.44	0.63	0.87	14.58
Jarboui et al. (2008)	0.03	0.09	0.36	0.44	0.89	1.1	18.14
Ranjbar and Kianfar (2009)	0.18	0.65	0.89	0.95	1.21	1.64	16.21
Alcaraz et al. (2003)	0.24	0.73	1.00	1.12	1.43	1.91	21.83
Józefowska et al. (2001)	1.16	1.73	2.6	4.07	5.52	6.74	18.64

Tableau 8: Déviation moyenne par rapport à l'optimum/à la longueur du chemin critique pour les instances de Kolisch et al. (1996).

	Instances				
	Boctor 50	MMLIB 50	Boctor 100	MMLIB 100	MMLIB+
Van Peteghem and Vanhoucke (2011)	23.79	25.45	25.11	26.51	101.45
Van Peteghem and Vanhoucke (2010)	23.41	27.12	24.67	29.55	97.59
Lova et al. (2009)	23.65	28.59	24.63	31.10	114.07
Boctor (1996b)	25.13	-	26.82	-	-
Olaguibel and Goerlich (1993)	26.52	-	29.16	-	-

Tableau 9: Déviation moyenne par rapport à la longueur du chemin critique pour les instances de Boctor (1996b) et celles de Van Peteghem and Vanhoucke (2014)

8 Problème d'ordonnancement de projet avec temps de transfert des ressources

Le POPRL classique, tel que défini précédemment, suppose qu'aucun temps de transfert n'est nécessaires pour transférer des ressources entre les différents lieux ou sites d'exécution des différentes tâches. En pratique, il arrive que le déplacement des ressources entre ces sites peut prendre un temps non-négligeable. Par exemple, le matériel de construction lourd pourrait être demandé à différents sites de construction et de les déplacer d'un site à l'autre peut nécessiter plusieurs heures.

Dans un tel cas, la modélisation et la résolution du problème d'ordonnancement de projet sans prendre en considération les temps de transfert pourrait produire des ordonnancements irréalisables. Récemment Krüger and Scholl (2009) ont développé des méthodes pour résoudre le problème d'ordonnancement de projet avec ressources limitées avec des temps de transfert (POPRLTT) dans un contexte d'un projet unique et pour un ensemble de projets. Les auteurs ont formulés les deux problèmes en faisant appel à la programmation linéaire. Des heuristiques constructives ont ensuite été proposées. Ensuite, Krüger (2009) a développé un algorithme génétique pour ce problème. Récemment, un algorithme de recherche tabu basé sur les flux a été développé par Poppenborg and Knust (2014). Kadri and Boctor (2014) ont étudié le problème quand les tâches peuvent avoir plusieurs modes d'exécution POPRL/PMETT.

9 Problème de minimisation des coût du projet

Dans la littérature certains chercheurs ont étudié le problème d'ordonnancemnt de projet dans l'objectif de minimiser le coût du projet. Ce problème est appelé problème de coût de disponibilité des ressources ou en anglais «Resource availability cost problem (RACP)». Dans ce problème, l'objectif est de déterminer les quantités de ressources à allouer au projet afin de minimiser une fonction de coût des ressources qui ne dépend pas de la durée du projet. Möhring (1984) a été le premier à introduire le RACP. Il a developé une procédure de résolution exacte sous l'hypothèse que les tâches ont un seul mode d'exécution. Drexl and Kimms (2001) ont développé deux bornes inférieures pour le problème en utilisant des techniques de relaxation Lagrangienne et de génération de colonne. Le problème qu'ils considéraient et appelé le problème d'investissement, ou en anglais «Resource Investement Problem (RIP)», est légèrement différent de la RACP car ils n'imposent pas une date due au projet. Des approches heuristiques sont

également proposées pour résoudre le RACP. Yamashita et al. (2006) a proposé une heursitique de recherche dispersée (Scatter search). Quant à Shadrokh and Kianfar (2007) ont proposé un algorithme génétique.

On constate que la plupart des approches de résolution développées sont conçues pour résoudre le problème dans un contexte de mode unique d'exécution des tâches. La seule heuristique conçue pour résoudre la version avec plusieurs modes d'exécution est celle de Hsu and Kim (2005). Les auteurs proposent une heursitique basée sur des règles de priorité. Cependant, pour résoudre ce problème, les chercheurs font une hypothèse très discutable. Ils supposent que le coût d'utilisation d'une unité de ressource, disponible à partir du début à la fin d'exécution du projet reste constant quelle que soit la durée du projet. Cette hypothèse simplifie considérablement la modélisation du problème et réduit la complexité des méthodes de résolution. Mais cette hypothèse est loin des situations pratiques où le coût de disponibilité des ressources est plus souvent proportionnel à la durée du projet. Aussi, nous constatons que dans les deux versions du RACP ou RIP, les frais généraux ne sont pas pris en compte. Encore une fois, dans la pratique, les frais généraux représente une partie importante du coût d'exécution du projet et ne doivent pas être négligés.

10 Conclusion

Dans ce document, on a revu le problème d'ordonnancement de projet à ressources limitées POPRL. Ce problème est devenu ces dernières années un problème de référence en gestion de projet (Hartmann (2012)). Il s'agit de déterminer les dates d'exécution des tâches dans l'objectif de minimiser la durée totale du projet, tout en respectant les contraintes de préséance entre les tâches du projet et les limites sur les disponibilités des ressources.

Le POPRL tel qu'il a été défini précédemment est une simplification des situations pratiques, ce qui a incité les chercheurs ces dernières années à proposer plusieurs généralisations du problème classique, tel que le problème d'ordonnancement de projet à ressources limitées où les tâches ont plusieurs modes d'exécution (POPRL/PME). Dans ce dernier chaque tâche a un nombre limité de modes d'exécution où chaque mode est caractérisé par les quantités des ressources renouvelables et non renouvelables requises et la durée d'exécution que l'on peut obtenir si ces quantités de ressources sont utilisées pour exécuter la tâche. Il s'agit non seulement de trouver les dates de début ou de fin des tâches, mais aussi de choisir le mode d'exécution à utiliser pour chaque

tâche avec l'objectif de minimiser la durée du projet. Le POPRL/PME a été d'une grande popularité ces dernières années, une revue de littérature a récemment été proposé par Węglarz et al. (2011). Ces problèmes ont fait l'objet de plusieurs travaux de recherche; des méthodes aussi bien exactes pour les petites instances, que heuristiques ont été proposées. Les algorithmes génétiques se distinguent parmi les techniques qui performent le mieux pour la résolution du POPRL et du POPRL/PME.

On constate que malgré les efforts déployés pour définir des POPRL plus généraux, les contraintes du temps de transfert des ressources continuent à être ignorées, nous constatons aussi que l'optimisation du problème en considérant les couts a été très peu traité dans la littérature. Cela force les gestionnaires dans la plus part des cas à se baser uniquement sur leur expérience pour réaliser ou ajuster manuellement les plans produits par des heuristiques conçues pour résoudre des versions simplifiées du problème.

11 Références

Abbasi, B., Shadrokh, S., Arkat, J., 2006. Bi-objective resource-constrained project scheduling with robustness and makespan criteria. Applied mathematics and computation 180 (1), 146–152.

Al-Fawzan, M. A., Haouari, M., 2005. A bi-objective model for robust resource-constrained project scheduling. International Journal of production economics 96 (2), 175–187.

Alcaraz, J., M, C., R, R., 2003. Solving the multi-mode resource-constrained project scheduling problem with genetic algorithms. Journal of the Operational Research Society 54 (6), 614–626.

Alvarez-Valdes, R., Tamarit, J. M., 1989. Heuristic algorithms for resource-constrained project scheduling: A review and an empirical analysis. In: Advances in project scheduling. Vol. 9. Elsevier Amsterdam, pp. 113–134.

Artigues, C., Michelon, P., Reusser, S., 2003. Insertion techniques for static and dynamic resource-constrained project scheduling. European Journal of Operational Research 149 (2), 249–267.

Baar, T., Brucker, P., S, K., 1999. Tabu search algorithm and lower bounds for the resource-constrained project scheduling problem. In: Voss et al. (eds) Meta heuristics : Advances and trends in local search paradigms for optimization. Springer, pp. 1–19.

Balas, E., 1985. On the facial structure of scheduling polyhedra. Mathematical programming studies, 179–218.

Ballestín, F., Valls, V., Quintanilla, S., 2008. Pre-emption in resource-constrained project scheduling. European Journal of Operational Research 189 (3), 1136–1152.

Bartusch, M., Möhring, R. H., Radermacher, F. J., 1988. Scheduling project networks with resource constraints and time windows. Annals of operations Research 16 (1), 199–240.

Bell, C. E., Han, J., 1991. A new heuristic solution method in resource-constrained project scheduling. Naval Research Logistics (NRL) 38 (3), 315–331.

Blazewicz, J., Lenstra, J. K., Kan, A., 1983. Scheduling subject to resource constraints: classification and complexity. Discrete Applied Mathematics 5 (1), 11–24.

Boctor, F. F., 1990. Some efficient multi-heuristic procedures for resource-constrained project scheduling. European Journal of Operational Research 49 (1), 3–13.

Boctor, F. F., 1993. Heuristics for scheduling projects with resource restrictions and several resource-duration modes. International Journal of Production Research 31 (11), 2547–2558.

Boctor, F. F., 1996a. A new and efficient heuristic for scheduling projects with resource restrictions and multiple execution modes. European Journal of Operational Research 90 (2), 349–361.

Boctor, F. F., 1996b. Resource-constrained project scheduling by simulated annealing. International Journal of Production Research 34 (8), 2335–2351.

Boctor, F. F., d'Avignon, G., 2005. A tabu search algorithm for the multiple mode resource-constrained project scheduling problem. Gestion des Opérations et Production, 28–89.

Boschetti, M., Maniezzo, V., 2009. Benders decomposition, lagrangean relaxation and metaheuristic design. Journal of Heuristics 15 (3), 283–312.

Bouleimen, K., Lecocq, H., 2003. A new efficient simulated annealing algorithm for the resource-constrained project scheduling problem and its multiple mode version. European Journal of Operational Research 149 (2), 268–281.

Brucker, P., Knust, S., Schoo, A., Thiele, O., 1998. A branch and bound algorithm for the resource-constrained project scheduling problem. European Journal of Operational Research 107 (2), 272–288.

Buddhakulsomsiri, J., Kim, D. S., 2006. Properties of multi-mode resource-constrained project scheduling problems with resource vacations and activity splitting. European Journal of Operational Research 175 (1), 279–295.

Calhoun, K. M., Deckro, R. F., Moore, J. T., Chrissis, J. W., Van Hove, J. C., 2002. Planning and re-planning in project and production scheduling. Omega 30 (3), 155–170.

Carlier, J., Latapie, B., 1991. Une méthode arborescente pour résoudre les problèmes cumulatifs. Revue française d'automatique, d'informatique et de recherche opérationnelle. Recherche opérationnelle 25 (3), 311–340.

Chen, R.-M., 2011. Particle swarm optimization with justification and designed mechanisms for resource-constrained project scheduling problem. Expert Systems with Applications 38 (6), 7102 – 7111.

Chiang, C.-W., Huang, Y.-Q., Wang, W.-Y., 2008. Ant colony optimization with parameter adaptation for multi-mode resource-constrained project scheduling. Journal of Intelligent & Fuzzy Systems 19 (4, 5), 345–358.

Christofides, N., Alvarez-Valdés, R., Tamarit, J. M., 1987. Project scheduling with resource constraints: A branch and bound approach. European Journal of Operational Research 29 (3), 262–273.

Chtourou, H., Haouari, M., 2008. A two-stage-priority-rule-based algorithm for robust resource-constrained project scheduling. Computers & industrial engineering 55 (1), 183–194.

Chyu, C.-C., Chen, A. H., Lin, X.-H., 2005. A hybrid ant colony approach to multi-mode resource-constrained project scheduling problems with non-renewable types. In: Proceedings of First International Conference on Operations and Supply Chain Management, Bali.

Coelho, J., Valadares, L. T., 2002. Comparative analysis on approximation algorithms for the resource constrained project scheduling problem. In: Eighth International Workshop on Project Management and Schduling. EURO Working Group on Project Management and Scheduling.

Coelho, J., Vanhoucke, M., 2011. Multi-mode resource-constrained project scheduling using rcpsp and sat solvers. European Journal of Operational Research 213 (1), 73–82.

Damak, N., Jarboui, B., Siarry, P., Loukil, T., 2009. Differential evolution for solving multi-mode resource-constrained project scheduling problems. Computers & Operations Research 36 (9), 2653–2659.

Damay, J., Quilliot, A., Sanlaville, E., 2007. Linear programming based algorithms for preemptive and non-preemptive rcpsp. European Journal of Operational Research 182 (3), 1012–1022.

De Reyck, B., et al., 1998. A branch-and-bound procedure for the resource-constrained project scheduling problem with generalized precedence relations. European Journal of Operational Research 111 (1), 152–174.

Debels, D., De Reyck, B., Leus, R., Vanhoucke, M., 2006. A hybrid scatter search/electromagnetism meta-heuristic for project scheduling. European Journal of Operational Research 169 (2), 638–653.

Demeulemeester, E., Herroelen, W., 1992. A branch-and-bound procedure for the multiple resource-constrained project scheduling problem. Management Science 38 (12), 1803–1818.

Demeulemeester, E. L., Herroelen, W. S., 1997. New benchmark results for the resource-constrained project scheduling problem. Management Science 43 (11), 1485–1492.

Demeulemeester, E. L., Herroelen, W. S., 2006. Project scheduling: a research handbook. Vol. 49. Springer Science & Business Media.

Doerner, K. F., Gutjahr, W. J., Hartl, R. F., Strauss, C., Stummer, C., 2008. Nature-inspired metaheuristics for multiobjective activity crashing. Omega 36 (6), 1019–1037.

Drexl, A., Kimms, A., 2001. Optimization guided lower and upper bounds for the resource investment problem. Journal of the Operational Research Society, 340–351.

El Sakkout, H., Wallace, M., 2000. Probe backtrack search for minimal perturbation in dynamic scheduling. Constraints 5 (4), 359–388.

Elloumi, S., Fortemps, P., Teghem, J., Loukil, T., 2006. A new bi-objective algorithm using clustering heuristics to solve the multi-mode resource-constrained project scheduling problem. In: Proceedings of the 25th workshop of the UK planning and scheduling special interest group (PlanSIG 2006), Nottingham. pp. 113–120.

Franck, B., Neumann, K., Schwindt, C., 2001. Project scheduling with calendars. OR-Spektrum 23 (3), 325–334.

Gomes, H. C., de Assis das Neves, F., Souza, M. J. F., 2014. Multi-objective metaheuristic algorithms for the resource-constrained project scheduling problem with precedence relations. Computers & Operations Research 44, 92 – 104.

Gonçalves, J. F., Resende, M. G. C., Mendes, J. J. M., 2011. A biased random-key genetic algorithm with forward-backward improvement for the resource constrained project scheduling problem. Journal of Heuristics 17 (5), 467–486.

Hapke, M., Jaszkiewicz, A., Słowiński, R., 1998. Interactive analysis of multiple-criteria project scheduling problems. European Journal of Operational Research 107 (2), 315–324.

Hartmann, S., 1998. A competitive genetic algorithm for resource-constrained project scheduling. Naval Research Logistics (NRL) 45 (7), 733–750.

Hartmann, S., 2002. A self-adapting genetic algorithm for project scheduling under resource constraints. Naval Research Logistics (NRL) 49 (5), 433–448.

Hartmann, S., 2012. Project scheduling under limited resources: models, methods, and applications. Vol. 478. Springer Science & Business Media.

Hartmann, S., Briskorn, D., 2010. A survey of variants and extensions of the resource-constrained project scheduling problem. European Journal of Operational Research 207 (1), 1–14.

Hartmann, S., Drexl, A., 1998. Project scheduling with multiple modes: a comparison of exact algorithms. Networks 32 (4), 283–297.

Herroelen, W., De Reyck, B., Demeulemeester, E., 1998. Resource-constrained project scheduling: a survey of recent developments. Computers & Operations Research 25 (4), 279–302.

Hindi, K. S., Yang, H., Fleszar, K., 2002. An evolutionary algorithm for resource-constrained project scheduling. Evolutionary Computation, IEEE Transactions on 6 (5), 512–518.

Hsu, C., Kim, D. S., 2005. A new heuristic for the multi-mode resource investment problem. Journal of the Operational Research Society 56 (4), pp. 406–413.

Icmeli-Tukel, O., Rom, W. O., 1997. Ensuring quality in resource constrained project scheduling. European Journal of Operational Research 103 (3), 483–496.

Jarboui, B., Damak, N., Siarry, P., Rebai, A., 2008. A combinatorial particle swarm optimization for solving multi-mode resource-constrained project scheduling problems. Applied Mathematics and Computation 195 (1), 299–308.

Jedrzejowicz, P., Ratajczak, E., 2006. Population learning algorithm for the resource-constrained project scheduling. Springer.

Józefowska, J., Mika, M., Różycki, R., Waligóra, G., Węglarz, J., 2001. Simulated annealing for multi-mode resource-constrained project scheduling. Annals of Operations Research 102 (1-4), 137–155.

Kadri, R. L., Boctor, F. F., 2014. Multi-mode resource constrained project scheduling with sequence dependent transfer times. In: Proceedings of the 14th International Conference on Project Management and Scheduling. pp. 106 – 109.

Khoshjahan, Y., Najafi, A. A., Afshar-Nadjafi, B., 2013. Resource constrained project scheduling problem with discounted earliness–tardiness penalties: Mathematical modeling and solving procedure. Computers & Industrial Engineering 66 (2), 293 – 300.

Kim, K., Yun, Y., Yoon, J., Gen, M., Yamazaki, G., 2005. Hybrid genetic algorithm with adaptive abilities for resource-constrained multiple project scheduling. Computers in industry 56 (2), 143–160.

Klein, R., 1999. Scheduling of resource-constrained projects. Vol. 10. Springer Science & Business Media.

Klein, R., Scholl, A., 1999. Computing lower bounds by destructive improvement: An application to resource-constrained project scheduling. European Journal of Operational Research 112 (2), 322–346.

Kobylański, P., Kuchta, D., 2007. A note on the paper by ma al-fawzan and m. haouari about a bi-objective problem for robust resource-constrained project scheduling. International Journal of Production Economics 107 (2), 496–501.

Kochetov, Y., Stolyar, A., 2003. Evolutionary local search with variable neighborhood for the resource constrained project scheduling problem. In: Proceedings of the 3rd international workshop of computer science and information technologies. Vol. 132.

Kohlmorgen, U., Schmeck, H., Haase, K., 1999. Experiences with fine-grainedparallel genetic algorithms. Annals of Operations Research 90, 203–219.

Kolisch, R., 1996. Serial and parallel resource-constrained project scheduling methods revisited: Theory and computation. European Journal of Operational Research 90 (2), 320–333.

Kolisch, R., 2000. Integrated scheduling, assembly area-and part-assignment for large-scale, make-to-order assemblies. International Journal of Production Economics 64 (1), 127–141.

Kolisch, R., Drexl, A., 1997. Local search for nonpreemptive multi-mode resource-constrained project scheduling. IIE transactions 29 (11), 987–999.

Kolisch, R., Hartmann, S., 2006. Experimental investigation of heuristics for resource-constrained project scheduling: An update. European Journal of Operational Research 174 (1), 23–37.

Kolisch, R., Sprecher, A., 1997. Psplib-a project scheduling problem library: Or software-orsep operations research software exchange program. European Journal of Operational Research 96 (1), 205–216.

Koulinas, G., Kotsikas, L., Anagnostopoulos, K., 2014. A particle swarm optimization based hyper-heuristic algorithm for the classic resource constrained project scheduling problem. Information Sciences 277, 680 – 693.

Kreter, S., Rieck, J., Zimmermann, J., 2016. Models and solution procedures for the resource-constrained project scheduling problem with general temporal constraints and calendars. European Journal of Operational Research 251 (2), 387 – 403.

Krüger, D., 2009. Multi-project scheduling with transfers. Ph.D. thesis, University of Jena, Germany.

Krüger, D., Scholl, A., 2009. A heuristic solution framework for the resource constrained (multi-) project scheduling problem with sequence-dependent transfer times. European Journal of Operational Research 197 (2), 492–508.

Lee, J.-K., Kim, Y.-D., 1996. Search heuristics for resource constrained project scheduling. Journal of the Operational Research Society, 678–689.

Leyman, P., Vanhoucke, M., 2015. A new scheduling technique for the resource–constrained project scheduling problem with discounted cash flows. International Journal of Production Research 53 (9), 2771–2786.

Leyman, P., Vanhoucke, M., 2016. Payment models and net present value optimization for resource-constrained project scheduling. Computers & Industrial Engineering 91, 139–153.

Lova, A., Tormos, P., Barber, F., 2006. Multi-mode resource constrained project scheduling: scheduling schemes, priority rules and mode selection rules. Inteligencia Artificial 30 (10), 69–86.

Lova, A., Tormos, P., Cervantes, M., Barber, F., 2009. An efficient hybrid genetic algorithm for scheduling projects with resource constraints and multiple execution modes. International Journal of Production Economics 117 (2), 302–316.

Maniezzo, V., Mingozzi, A., 1999. The project scheduling problem with irregular starting time costs. Operations Research Letters 25 (4), 175–182.

Mendes, J. J. d. M., Gonçalves, J. F., Resende, M. G., 2009. A random key based genetic algorithm for the resource constrained project scheduling problem. Computers & Operations Research 36 (1), 92–109.

Messelis, T., De Causmaecker, P., 2014. An automatic algorithm selection approach for the multi-mode resource-constrained project scheduling problem. European Journal of Operational Research 233 (3), 511–528.

Mika, M., Waligora, G., Weglarz, J., 2005. Simulated annealing and tabu search for multi-mode resource-constrained project scheduling with positive discounted cash flows and different payment models. European Journal of Operational Research 164 (3), 639–668.

Mika, M., Waligora, G., Węglarz, J., 2008. Tabu search for multi-mode resource-constrained project scheduling with schedule-dependent setup times. European Journal of Operational Research 187 (3), 1238–1250.

Mingozzi, A., Maniezzo, V., Ricciardelli, S., Bianco, L., May 1998. An exact algorithm for the resource-constrained project scheduling problem based on a new mathematical formulation. Management Science 44 (5), 714–729.

Möhring, R. H., 1984. Minimizing costs of resource requirements in project networks subject to a fixed completion time. Operations Research 32 (1), 89–120.

Möhring, R. H., Schulz, A. S., Stork, F., Uetz, M., 2001. On project scheduling with irregular starting time costs. Operations Research Letters 28 (4), 149–154.

Montoya-Torres, J. R., Gutierrez-Franco, E., Pirachicán-Mayorga, C., 2010. Project scheduling with limited resources using a genetic algorithm. International Journal of Project Management 28 (6), 619–628.

Mori, M., Tseng, C. C., 1997. A genetic algorithm for multi-mode resource constrained project scheduling problem. European Journal of Operational Research 100 (1), 134–141.

Neumann, K., Zhan, J., 1995. Heuristics for the minimum project-duration problem with minimal and maximal time lags under fixed resource constraints. Journal of Intelligent Manufacturing 6 (2), 145–154.

Nonobe, K., Ibaraki, T., 2002. Formulation and tabu search algorithm for the resource constrained project scheduling problem. In: Essays and surveys in metaheuristics. Springer, pp. 557–588.

Nudtasomboon, N., Randhawa, S. U., 1997. Resource-constrained project scheduling with renewable and non-renewable resources and time-resource tradeoffs. Computers & Industrial Engineering 32 (1), 227–242.

Olaguíbel, R. A.-V., Goerlich, J. T., 1993. The project scheduling polyhedron: Dimension, facets and lifting theorems. European Journal of Operational Research 67 (2), 204 – 220.

Olaguibel, R. A.-V., Goerlich, J. T., 1993. The project scheduling polyhedron: dimension, facets and lifting theorems. European Journal of Operational Research 67 (2), 204–220.

Padman, R., Zhu, D., 2006. Knowledge integration using problem spaces: A study in resource-constrained project scheduling. Journal of Scheduling 9 (2), 133–152.

Patterson, J., Slowinski, R., Talbot, F., Weglarz, J., 1989. An algorithm for a general class of precedence and resource constrained scheduling problems. Advances in project scheduling 187, 3–28.

Patterson, J. H., 1984. A comparison of exact approaches for solving the multiple constrained resource, project scheduling problem. Management Science 30 (7), 854–867.

Pollack-Johnson, B., 1995. Hybrid structures and improving forecasting and scheduling in project management. Journal of Operations Management 12 (2), 101–117.

Poppenborg, J., Knust, S., 2014. A flow-based tabu search algorithm for the rcpsp with transfer times. In: Proceedings of the 14th International Conference on Project Management and Scheduling, March 30th– April 2nd 2014. pp. 181 – 184.

Pritsker, A. A. B., Waiters, L. J., Wolfe, P. M., 1969. Multiproject scheduling with limited resources: A zero-one programming approach. Management Science 16 (1), 93–108.

Ranjbar, M., Kianfar, F., 2009. A hybrid scatter search for the rcpsp. Scientia Iranica. Transaction E, Industrial Engineering 16 (1), 11.

Sampson, S. E., Weiss, E. N., 1993. Local search techniques for the generalized resource constrained project scheduling problem. Naval Research Logistics (NRL) 40 (5), 665–675.

Shadrokh, S., Kianfar, F., 2007. A genetic algorithm for resource investment project scheduling problem, tardiness permitted with penalty. European Journal of Operational Research 181 (1), 86 – 101.

Sprecher, A., 2000. Scheduling resource-constrained projects competitively at modest memory requirements. Management Science 46 (5), 710–723.

Sprecher, A., Hartmann, S., Drexl, A., 1997. An exact algorithm for project scheduling with multiple modes. Operations-Research-Spektrum 19 (3), 195–203.

Stinson, J. P., Davis, E. W., Khumawala, B. M., 1978. Multiple resource–constrained scheduling using branch and bound. AIIE Transactions 10 (3), 252–259.

Talbot, F. B., 1982. Resource-constrained project scheduling with time-resource tradeoffs: The nonpreemptive case. Management Science 28 (10), 1197–1210.

Thomas, P. R., Salhi, S., 1998. A tabu search approach for the resource constrained project scheduling problem. Journal of Heuristics 4 (2), 123–139.

Toklu, Y. C., 2002. Application of genetic algorithms to construction scheduling with or without resource constraints. Canadian Journal of Civil Engineering 29 (3), 421–429.

Tseng, L.-Y., Chen, S.-C., 2009. Two-phase genetic local search algorithm for the multimode resource-constrained project scheduling problem. Evolutionary Computation, IEEE Transactions on 13 (4), 848–857.

Valls, V., Ballestín, F., Quintanilla, S., 2004. A population-based approach to the resource-constrained project scheduling problem. Annals of Operations Research 131 (1-4), 305–324.

Valls, V., Ballestin, F., Quintanilla, S., 2008. A hybrid genetic algorithm for the resource-constrained project scheduling problem. European Journal of Operational Research 185 (2), 495–508.

Van de Vonder, S., Demeulemeester, E., Herroelen, W., 2007. A classification of predictive-reactive project scheduling procedures. Journal of Scheduling 10 (3), 195–207.

Van Peteghem, V., Vanhoucke, M., 2010. A genetic algorithm for the preemptive and non-preemptive multi-mode resource-constrained project scheduling problem. European Journal of Operational Research 201 (2), 409–418.

Van Peteghem, V., Vanhoucke, M., 2011. Using resource scarceness characteristics to solve the multi-mode resource-constrained project scheduling problem. Journal of Heuristics 17 (6), 705–728.

Van Peteghem, V., Vanhoucke, M., 2014. An experimental investigation of metaheuristics for the multi-mode resource-constrained project scheduling problem on new dataset instances. European Journal of Operational Research 235 (1), 62–72.

Vanhoucke, M., 2010. A scatter search heuristic for maximising the net present value of a resource-constrained project with fixed activity cash flows. International Journal of Production Research 48 (7), 1983–2001.

Vanhoucke, M., Coelho, J., Debels, D., Maenhout, B., Tavares, L. V., 2008. An evaluation of the adequacy of project network generators with systematically sampled networks. European Journal of Operational Research 187 (2), 511–524.

Vanhoucke, M., Demeulemeester, E., Herroelen, W., 2001. On maximizing the net present value of a project under renewable resource constraints. Management Science 47 (8), 1113–1121.

Viana, A., de Sousa, J. P., 2000. Using metaheuristics in multiobjective resource constrained project scheduling. European Journal of Operational Research 120 (2), 359–374.

Węglarz, J., Józefowska, J., Mika, M., Waligóra, G., 2011. Project scheduling with finite or infinite number of activity processing modes–a survey. European Journal of Operational Research 208 (3), 177–205.

Xiao, J., Wu, Z., Hong, X.-X., Tang, J.-C., Tang, Y., 2016. Integration of electromagnetism with multi-objective evolutionary algorithms for rcpsp. European Journal of Operational Research 251 (1), 22–35.

Yamashita, D. S., Armentano, V. A., Laguna, M., 2006. Scatter search for project scheduling with resource availability cost. European Journal of Operational Research 169 (2), 623 – 637.

Zamani, R., 2013. A competitive magnet-based genetic algorithm for solving the resource-constrained project scheduling problem. European Journal of Operational Research 229 (2), 552 – 559.

Zhang, H., Tam, C., Li, H., 2006. Multimode project scheduling based on particle swarm optimization. Computer-Aided Civil and Infrastructure Engineering 21 (2), 93–103.

Zhu, G., Bard, J. F., Yu, G., 2006. A branch-and-cut procedure for the multimode resource-constrained project-scheduling problem. INFORMS Journal on Computing 18 (3), 377–390.